МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему:

«Інформаційна технологія аналізу мережевого трафіку у локальних мережах з підтримкою мультимедійних сервісів»

Завідувач випускаючої кафедри — **Довбиш А.С.**

Керівник роботи — **Великодний Д.В.**

Студента групи ІН.м.-91н — **Тітарєв А.О.**

СУМИ 2021

Сумський державний університет

(назва вузу)

Факультет *ЕлІТ* Кафедра *Комп'ютерних наук*

Спеціальність *«Інформатика»*

Затверджую:

зав.кафедрою _____

_____

"_____" _____ 20___р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТОВІ

Тітарєву Антону Олександровичу

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) *Інформаційна технологія аналізу мережевого трафіку у локальних мережах з підтримкою мультимедійних сервісів*

затверджую наказом по інституту від "____" _____ 20 ___ р. № _____

2. Термін здачі студентом закінченого проекту (роботи)_____

3. Вхідні данні до проекту (роботи) _____
_____
_____
_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Літературний огляд та постановка задачі. 2) Моделювання технології передачі медіаданих з використанням методології QoS та аналіз пакетів трафіку.3) Створення схеми IPTV з методологією QoS та розробка графічного інтерфейсу. 4) Тестування інтерфейсу.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) _____
_____
_____
_____
_____

6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання _____

Керівник _____

(підпис)

Завдання прийняв до виконання _____

(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проекту (роботи)	Термін виконання проекту (роботи)	Примітка
1.	Аналіз проблеми предметної області, постановка й формування завдань дослідження		
2.	Огляд технологій, що використовуються для визначення змін в потокових відео		
3.	Розробка інтелектуальної системи з застосуванням оптимізації методом зонування		
4.	Аналіз отриманих результатів		
5.	Оформлення пояснювальної записки до кваліфікаційної магістерської роботи		

Студент – дипломник _____

(підпис)

Керівник проекту _____

(підпис)

РЕФЕРАТ

Кваліфікаційна робота магістра містить 52 сторінки, 1 таблицю, 22 рисунки, список літератури містить 21 найменування, 1 додаток.

Об’єкт дослідження – технологія мультимедійного трафіку у локальних мережах.

Мета роботи – розробка графічного інтерфейсу для налаштування інформаційної технології IPTV з методологією QoS у локальних ком’ютерних мережах.

Методи дослідження – моделювання у графічному емуляторі комп'ютерних мереж GNS3 та аналіз трафіку за допомогою програми-сніфера.

Результати – розроблено веб-орієнтовний графічний інтерфейс для генерування скриптів, за допомого яких можливо налаштувати конфігурацію технології багатомовного транслювання відеопотоку IPTV з технологією QoS. Це рішення дозволяє оптимізувати процес налаштування локальної мережі та пришвидшує процес її аналізу та розгортання на реальному обладнанні. Інтерфейс реалізовано у формі веб-додатку з використанням мови JavaScript.

МУЛЬТИМЕДІА, MULTICAST, МАРШРУТИЗАТОР, IPTV, IGMP, PIM, QOS, СНІФФЕР, GNS3, CISCO, JAVASCRIPT

ЗМІСТ

ВСТУП

На сьогоднішній день сфера телекомунікацій є однією з найпрогресивних та найпопулярніших у всьому світі, адже перед наю стоїть дуже важлива задача – об’єднати увесь світ у одну цілу мережу, у якій не важливо де і хто знаходиться, але всі можуть обмінюватися інформацією. Однією з прогресивних технологій поширення інформації є IPTV, яка повністю замінює аналогове чи супутникове телебачення через свою гнучкість та підхід до користувача.

Завданням наукової роботи обрано інформаційну систему для написання налаштування мережі. На сьогоднішній день аналогів такого сервісу дуже мало, адже такими налаштуваннями займаються в основному провайдери. Метою роботи є огляд відомих технологій, глибокий аналіз мережі з підключеною технологією, покращення якості надання послуг та реалізація інформаційної системи, яка виступатиме у ролі головного інструменту для налаштування комп’ютерної мережі з технологією IPTV та застосованою методологією QoS. Задача такого інструменту в першу чергу полегшити та пришвидшити процес моделювання, проектування та розгортаняя мережі.

Для розробки було обрано мову програмування JavaScript та середовище для емуляції комп’ютерної мережі GNS3. У рамках роботи необхідно вивчити предметну область, дослідити існуючі підходи, проаналізувавши сильні та слабкі сторони. Головним підходом до реалізації даної роботи є початкове проектування реальної мережі у симуляторі, застосування методології QoS та технології IPTV, аналіз отриманої мережі за допомогою програми-сніфера та реалізація готової схеми у програмі.

1 ЛІТЕРАТУРНИЙ ОГЛЯД

1.1 Технологія IPTV

Телевізійний Інтернет протокол чи, іншими словами, IPTV – це технологія відео-мовлення, що переправляю відео-контент у комп'ютерних мережах. Замість використання звичних «посередників» телевізійних кабелів чи супутникових антен, у технології IPTV використовуються об'єднані IP мережі, що в свою чергу дає доступ до інших сервісів так, як: VoIP (IP-телефонія), обмін комп'ютерними даними чи навіть звичайний доступ до Інтернету усередині однієї інтернет-мережі.

Із самого початку технологоія IPTV була доступною лучше на комп'ютерах та телевізорах. На сьогоднішній день ця технологія набула популярності серед гравців у веб-ігри, мобільні додатки запровадили Live TV та технологію «Відео-на-замовлення» у багатьох пристроях. Ця добре відома технологія вже добре стоїть на ногах і надалі буде рушійною силою, яка визначає досвід глядачів, оскільки все більше провайдерів, домогосподарів та готелів усе більше і більше можернізують свою телевізійні системи [1].

За самою технологією стоїть процес трансляції відео-потоку від постачальника до самого користувача. Сьогодні набуває неабиякої популярності сервіс «Відео-на-вимогу» (VOD). Головною перевагою цього сервісу навідміну від звичайного аналоговогу чи супутникового телебачення є те, що кінцевий користувач не прив'язаний до трансляції, він має змогу дивитися те, що хоче зараз переглянути. Це відбувається за рахунок надісланого запиту до постачальника і він одразу почне транслювати те, що користувач захоче. Система відео по замовленню має багато різновидів:

- Near video on demand (nVOD) – найбільш популярний вид. У цьому випадку оператор заздалегідь генерує чи готує програму для показу, а вже потім розсилає контент по мірі надходження запитів від користувачів

– Push video on demand (pVOD) – не менш популярний сервіс, що використовується у випадках, коли пряма трансляція має недоліки. Тоді передача та відео-контент записується поступово на приймаючий пристрій, а вже потім транслюється
– True video on demand (tVOD) – «справжнє» відео по замовленню, коли користувач отримує відео-потік, що був сформований заздалегідь спеціально для нього [2].

Архітектурно комплекс IPTV складається за таких підсистем, як:

- управління комплексом та послугами – ця система ще називається проміжковим програмним забезпеченням
- прийом та обробка контента
- захист контенту
- відеосервер
- контроль якості потоків трафіку

У якості клієнтського оснащеня може виступати відповідний до системних потреб комп'ютер, спеціалізована приставка-прийомник, телевізор з технологією SMART TV або мобільний пристрій. На програмному рівні доступ до ресурсів IPTV здійснюється за допомогою спеціальних програм чи додатків, а також за домогою вбудованого чи інтегрованого у пристрій інтернет-браузеру. Доставка від серверу до клієнту здійснюється по керованій IP-мережі за допомогою технології Multicast чи unicast (групова передача чи єдиному адресату).

Щодо технології багатоадресного чи точкового надсилання трафіку – це залежить напряму від топології мережі. Доцільно застосовувати технологію Multicast у випадку, коли джерело потоку відео-трафіку знаходиться далеко від приймачів сигналу. Також вигідно використати багатоадресне мовлення, щоб позбавити джерело потоку (відеосервер чи відеокамера) задачі визначати кількість приймачів та генерувати відповідну кількість копій відеопотоку до кожного з них.

1.2 Протоколи IGMP та PIM

Протокол IGMP був розроблений у 1989 році для покращення ефективності розсилки інформації по IP-адерсам і по сьогоднішній день він використовується виключно для взаємодії пов’язаних безпосередньо друг з другом маршрутизатора та так званого хоста, коли сам хост являється отримувачем трафіку групового мовлення [3]. Сьогодні більш популярним є протокол IGMPv2, він підтримується у багатьох сучасних операційних системах. У цьмогу протоколі визначено такі типи повідомлень:

- Membership query – запит про те, чи належить хост до певної групи мережі. Запит від маршрутизатора може бути загальний, щоб дізнатися про усі групи у мережі чи прицільний, щоб дізнатися про конкретну групу, адреса якої визначається у тілі запиту.
- Membership report – звіт від хоста про належність до групи. Є повідомленням-відповіддю на запит маршрутизатора. Текст повідомлення містить адресу групи, до якою сам хост належить. Цей звіт може бути надісланий хостом не лише, як відповідь на запит, але і з власної ініціативи. Це відбувається, коли хост хоче стати членом якоїсь певної групи.
- Leave group – повідомлення говорить саме за себе і свідчить про покидання групи. Воно необхідне для того, щоб хост міг сповістити маршрутизатор про те, що він покинув певну групу мережі. Маршрутизатор у свою чергу, отримавши це повідомлення, надсилає відповідний запит на цю групу, щоб перевірити чи є хост, що покидаю групу – останнім. Якщо так, то маршрутизатор перестає надсилати трафік загального мовлення на цю групу.

Усі ці повідомлення потрібні лише для того, щоб маршрутизатори визначали, до якої групи мережі потрібно надсилати трафік, а до якої – ні.

На Рисунку 1.1 представлена структура повідомлення для версії IGMPv2

Рис. 1.1 – структура повідомлення протоколу IGMP [3]

Незалежна від протоколу багатоадресна передача (PIM) – це набір протоколів багатоадресної маршрутизації, кожен з яких оптимізований для різних середовищ. Існує два основних проколи PIM: PIM sparse-mode (розрідженний режим) та PIM dense-mode (щільний режим). Третій протокол – змішаний, він малорозповсюджений. Розрідженний протокол – це протокол багатоадресної маршрутизації, який створений для того, щоб одержувачі конкретної групи розсилки будуть «рідко» розподілені по мережі. Іншими словами, припускається, що більшість підмереж в мережі не хочуть жодного багатоадресного пакету трафіку. Важливаю вимогою розрідженного режиму PIM є потреба у створенні адреси RP (точка рандеву) для групи багатоадресної передачі ща допомогою спільного дерева [4]. Основною перевагою перед dense-mode є те, що немає необхідності транслювати трафік і перенаправляти його, доки хост сам не попросить. Але у цей самий час, виникає проблема у тому, що групи, які не транслюють трафік можуть не дізнатися, хто саме його транслює. Для вирішення цієї проблеми і існує точка рандеву. Вона створюється на одному с роутерів та виступає за місце зустрічі. Маршрутизатор, що транслює трафік, надсилає його до точки рандеву, а роутер, що хочу отримувати трафік, забирає його з точки рандеву [5].

Протокол PIM dense-mode навпаки розроблений з припущенням, що приймачі багатоадресної передачі щільно роподілені по мережі. Мається на увазі те, що більшість хостів хочуть отримувати трафік, тому групове мовлення здійснюється на всю мережу, а вже у свою чергу маршрутизатори, у яких не має зацікавлених хостів, надсилають повідомлення про те, щоб видатит себе з дерева. Цей протокол є

простішим, так як не потребує розгортання точки рандеву, але для раціональності використання потребує, щоб більшість хостів потребувала трафік, щоб трафік був потрібний та не засмічував мережу.

1.3 Методологія QoS

Quality of Service – це методика використання механізмів або технологій для керування трафіком та забезпечення роботи важливих чи критично важливих систем або додатків. Методологія дозволяє організаціям налаштовувати загальний мережевий трафік, надаючи пріоритет високопродуктивним програмам, покращуючи якість обслуговування мережі.

Частіше за все, QoS використовують у мережах, які несуть трафік для ресурсоємних рішень. Загальні послуги, які потребують покращення за методологією QoS, включають телебачення через Інтернет (IPTV), онлайн-ігри, потокова передача медіа та мултимедіа, відеоконференції або передача голосу через IP (VoIP). Ключовою метою QoS є надання мережам та організаціям пріоритетності трафіку, що включає виділення особливої смуги пропускання трафіку, контролю джиттера та меншої затримки. Технології, що використовуються для цього, є життєво важливими для підвищення ефективності бізнес-додатків, широкосмугових мереж (WAN) та мереж постачальників послуг.

Здатність мережі забезпечити різні рівні обсулуговування, запитані різними мереживими додатками може бути класифікована трьома категоріями:

- Негарантована доставка. Вузли мережі зв’язані, але без гарантії часу та самого факту доставки пакетів трафіку у точку призначення. Насправді ця категорія не є частиною QoS через відсутність гарантії якості обслуговування. Але, хоча продуктивність передачі даних зменшена, для більшості додатків, орієнтованих на передачу інформації (наприклад додатки на протоколах FTP), така послуга є більш ніж достатньою.

- Дифернційоване обслуговування. Така категорія пропонує розділення трафіку на класи на основі вимог до якості послуг. Цей вид QoS зручно застосовувати до мережей з інтенсвним трафіком додатків. У цьому випадку зручно відділяти адміністративний чи критично-важливий трафік від іншого, назначивши йому пріоритет.
- Гарантоване обслуговування. Являє собою резервування мережевих ресурсів, щоб забезпечити якість та покрити специфічні вимоги до обслуговування зі сторони потоків трафіку. На жаль, забезпечити жорсткий QoS неможливо у масштабах магістралей Internet, яка обслуговує тисячі потоків трафіку одночасно, для цього створена агрегована резервація ресурсів, яка потребую збереження лише невеликої кількості інформації лише на базових маршрутизаторах.

Таблиця 1.1 – Рівні обслуговування та відповідні їм функції QoS [6]

Рівень обслуговування	Функція QoS мережевого рівня	Функція QoS канального рівня
Негарантована доставка	Зв’язність вузлів мережі	Технологія асинхронної передачі даних, обслуговування з неточною бітовою швидкістю, механізм узгодження швидкості передачі інформації
Дифернційоване обслуговування	Механізм узгодження швидкості доступу, алгоритм рівномірного обслуговування черг, алгоритм довільного раннього обслуговування	IEEE 802.1p

Гарантоване обслуговування	Протокол резервації ресурсів	Диспетчер пропускної здатності підмережі, механізм обслуговування з постійною бітовою швидкістю, механізм узгодження швидкості передачі інформації

Для покращення якості обслуговування використовують найчастіше такі прийоми, як: пріоритетність затримкового VoIP-трафіку через маршрутизатори та комутатори, резервування ресурсів, черги та позначення трафіку. А також відомі такі «кращі практики»: мінімазація складності конфігурації, покладання гарантій пропускної здатності лише на конкретні послуги для уникнення використання всього трафіку однією черго, наштування пріоритетності здійснюється лише на основі послуг або політики безпеки, не обох одночасно [7].

1.4 Огляд та аналіз програм-сніферів

Сніфери – це програми, які перехоплюють весь мережевий трафік. Головною задачею є діагностика мережі та її аналіз. Також такі програми можуть бути використані з ціллю перехоплення паролів чи несанкціонованого доступу[8]. Найчастіше ці програми використовують все ж таки системні адміністратори. За їх допомогою виявляються вади мережі чи проблемні місця. У локальній мережі можливо перехопити усі пакети трафіку з усіх машин. У широкомовній мережі всі вузли можуть отримувати всі повідомлення через єдину магістраль.

На сьогоднішній день аналізаторів мережевого трафіку існує дуже багато і використовуються вони на різних операційних системах. Частіше за все програми-сніфери розробляються мовами високого рівня (C, C++, Java тощо). Умовно аналізтори розподіляють на ті, що підтримують завантаження та роботу з командного рядка, і сніфери, що мають графічниййй інтерфейс. Також сніфери різняться між

собою глибиною аналізу мережі, підтримуваними протоколами та сумісністю з іншими платформами та програмами. Використовуються аналізатори як у деструктивних цілях, так і з добрими намірами. За допомогою сніффера можна вичислити ворожий паразитний чи вірусний трафік, несанкціоноване програмне забезпечення , локалізувати несправність, ало водночас і перехопити паролі, незашифрований (або іноді і зашифрований) трафік з ціллю крадіжки інформації або персональних даних. Для протидії ворожим намірам хакерів, можна знизити загрозу сніфферінгу за допомогою криптографії – особливого шифрування даних, або використовуючи спеціальні антисніффери – програми, що відстежують роботу програм-сніфферів та перешкоджають перехопленню інформації в мережі.

Розглянемо два популярних аналізатори трафіку: WireShark та SolarWinds NPS. Обидві програми належать до категорії «Моніторинг мережі» у технологічному стеці.

Деякі функції, пропоновані SolarWinds NPS:

- Моніторинг багатопрофільних мереж
- Мережева статистика для глибшої видимості
- Інтелектуальні карти

З іншого боку, Wireshark надає такі ключові функції:

- Глибока перевірка сотень протоколів, причому постійно додається більше
- Захоплення в режимі реального часу та аналіз у режимі офлайн
- Стандартний трипанельний браузер пакетів [9]

Wireshark - це передовий і широко використовуваний аналізатор мережевих протоколів. Ця програма-сніфер є дуже популярною серед системних адміністраторів, вона є кросплатформенною та має дуже багато програм, з якими вона може бути інтегрована.

Перевагою аналога від компанії SolarWinds є те, що зібрані дані з безперервних потоків мережевого трафіку та цифри аналізу можуть бути оброблені та представлені

у вигляді графіків, діаграм та таблиць, за допомогою яких з легкістю можна визначити, як саме використовується корпоративна мережа, ким та з якою метою.

На рисунку 1.2 та рисунку 1.3 представлено графічний інтерфейс обох програм

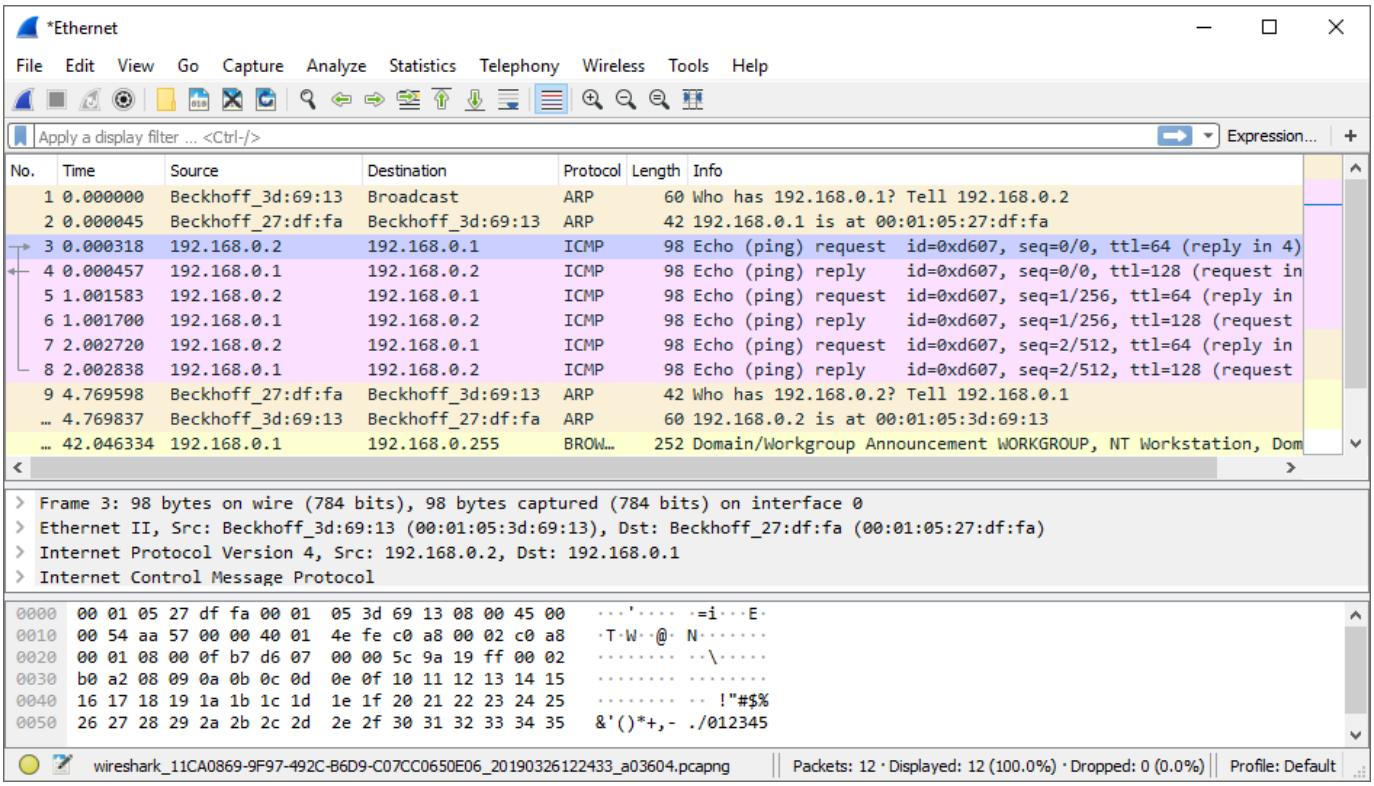

Рисунок 1.2 – Графічний інтерфейс програми WireShark

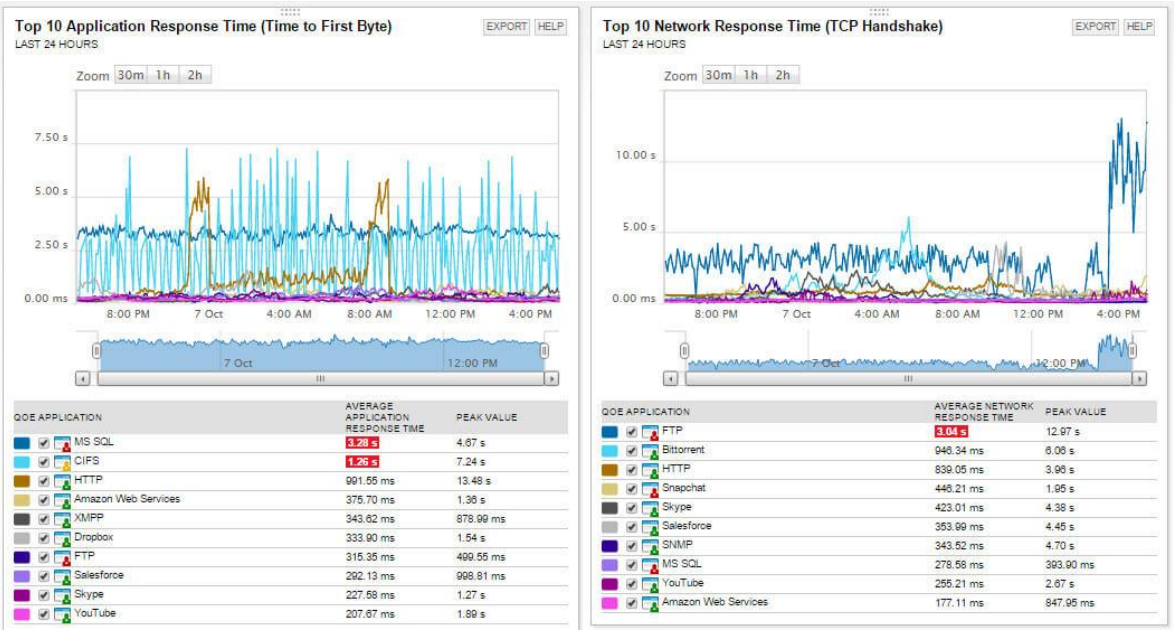

Рисунок 1.3 – Графічний інтерфейс програми SolarWinds Sniffer

1.5 Постановка задачі

Проаналізувавши зібрані дані, можна сформувати мету наступним чином: необхідно налаштувати мережевий інтерфейс роутерів з використанням IPTV за протоколами IGMP та PIM, налаштувати якість обслуговування для цієї мережі, проаналізувати трафік у цій мережі та розробити веб-орієнтований интерфейс, який буде дозволяти налаштовувати маршрутизатори зі вказаними технологіями. Важливо, щоб інформаційна система мала змогу генерувати та дозволяти копіювати набір команд для налаштування реального обладнання або симулятора мережевого трафіку. Програмне забезпечення повинно бути простим та зрозумілим кінцевому користувачу, з інтуїтивним інтерфейсом для тих, хто не знайомий зі схожими інтерфейсами та не має навичок конфігурації роутерів Cisco.

У якості графічного інтерфейсу повинна виступати веб-орієнтована інформаційна система, у яку можна занести вхідні дані про адреси маршрутизаторів, а в результаті отримати сгенеровані скрипти з налаштування обладнання Cisco.

Постановка задачі:

1. Моделювання та конфігурація мережі у симуляторі на базі обладнання Cisco
2. Аналіз зконфігурованої мережі за допомогою програми-сніффера
3. Розробка веб-орієнтованох інформаційної системи з графічним інтерфейсом для генерації скриптів налаштування роутерів
4. Тестування розробленого графічного інтерфейсу та згенерованих скриптів у симуляторі

2 МОДЕЛЮВАННЯ ТЕХНОЛОГІЇ ПЕРЕДАЧІ МЕДІАДАНИХ З ВИКОРИСТАННЯМ МЕТОДОЛОГІЇ QOS ТА АНАЛІЗ ПАКЕТІВ ТРАФІКУ У СЕРЕДОВИЩІ GNS3

2.1 Конфігурація у графічному мережевому симуляторі GNS3

Щоб сконфігурувати справжню мережу з використанням багатьох ресурсів, маршрутизаторів, контроллів та іншого допоміжного обладнання, потребується немало часу, але все це буде марно, якщо під час налаштування якийсь критичний момент буде пропущенно. Мережа може не працювати, не буде доступу до Інтернету або просто у якогось комп'ютера не буде доступний принтер, але дрібна помилка у конфігурації може призвести і до критичної поломки, яка буде коштувати дуже дорого. Ніхто не виключає людскої помилки під час налаштування мережі, але щоб зменшити ризик до мінімуму, мережу зпочатку необхідно налаштувати у симуляторі і лише після упевненості у правильності своїх дій, повної перевірки симулюючої схеми можна приступати до налаштування справжнього обладнання.

Сфера телекомунікацій неабияка популярна сьогодні і продовжує набирати обертів і тому створення та налаштування мережей Інтернет дуже часта річ. На озброєнні системних адміністраторів та телеком-спеціалістів існує безліч програм, симуляторів, емуляторів та додатків для створення, налаштування та емуляції комп'ютерної мережі для подальшого викорстання чи налаштування на живому обладнанні. До таких програм відноситься Cisco Packet Tracer, Boson NetSim, GNS3, VIRL та інші. Якщо Packet Tracer вважається «золотим стандартом» серед симуляторів віртуальних мереж, то GNS3 – серед емуляторів віртуальних мереж [11]. Інші додатки мають низку недоліків, серед яких і ціна за продукцію.

Для налаштування та досягення мети роботи середовища GNS3 достатньо, адже ця програма володіє усіма необхідними сервісами, являється кросплатформенною, має зручний графічний інтерфейс, що дає змогу з легкістю змоделювати майбутню мережу, встановши необхідні параметри на обраних у програмі пристроях. Також

плюсом середовища GNS3 є велика кількість інтеграцій з додатковими програмами, які необхідні при налаштуванні віртуальних машин (VirtualBox, VMware), інтегрований аналізатор трафіку WireShark. У останніх версіях програми з'явилася можливість додати віртуальну машину, яка може замінити комп'ютер на себе, тим самим усе навантаження від роутерів та створених пристроїв буде приходитися на віртуальну машину і використовувати її ресурс. Програма є безкоштовною та має відкритий доступ. Недоліком програми є лише те, що нееобхідно створювати власні образи мережевих пристроїв для емуляції. Це не провина програми, адже розробник просто не має доступу до інтеграції образів, власником яких є корпорація Cisco.

2.2 Налаштування технологій IPTV за допомогою середовища GNS3

Щоб розробити веб-орієнтовану інформаційну систему з налаштуванням технології IPTV, зпочатку необхідно змоделювати схему мережі у симуляторі GNS3. Примітивна схема IPTV включає в себе сервер (Multicast Server), який поширює мультикастовий трафік, клієнт (Multicast Client), що отримує трафік, та зв'язуючий маршрутизатор Cisco 7200. На рисунку 2.1 приклад такої схеми.

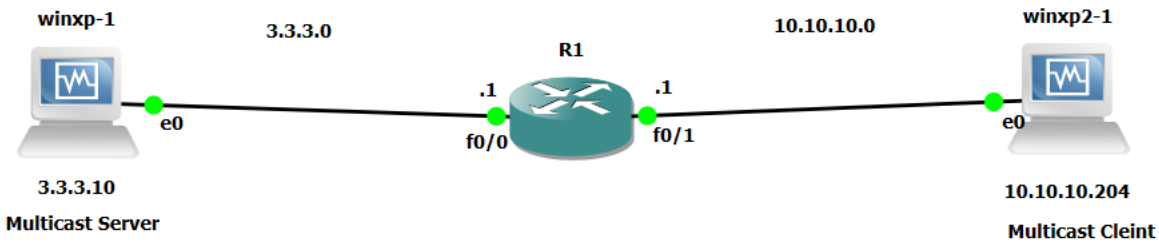

Рисунок 2.1 – Модель мережі з IPTV

Для емуляції джерелом та отримувачем багатоадресного трафіку використаємо віртуальні машини на базі Windows XP. Для мовлення трафіку використаємо VLC media player.

Зпочатку налаштування роутеру за допомогою консолі:

R1(config)#ip multicast-routing //вмикання мультикаст маршрутизації

R1(config)#int fa 0/0

R1(config-if)#ip address 3.3.3.1 255.255.255.0 // налаштування порту 0/0

R1(config-if)#ip pim sparse-mode // включення режиму Sparse-mode

R1(config-if)#no sh

R1(config-if)#exit

R1(config)#int fa 0/1

R1(config-if)#ip address 10.10.10.1 255.255.255.0 // налаштування порту 0/1

R1(config-if)#ip pim sparse-mode

R1(config-if)#no sh

R1(config-if)#exit

R1(config)#ip pim rp-address 3.3.3.1 // включення точки рандеву

Наступним кроком є налаштування локальних адрес для віртуальних машин. Для мультикаст серверу вказуємо адресу комп'ютера 3.3.3.10 255.255.255.0 з основним шлюзом 3.3.3.1 (Рисунок 2.2)

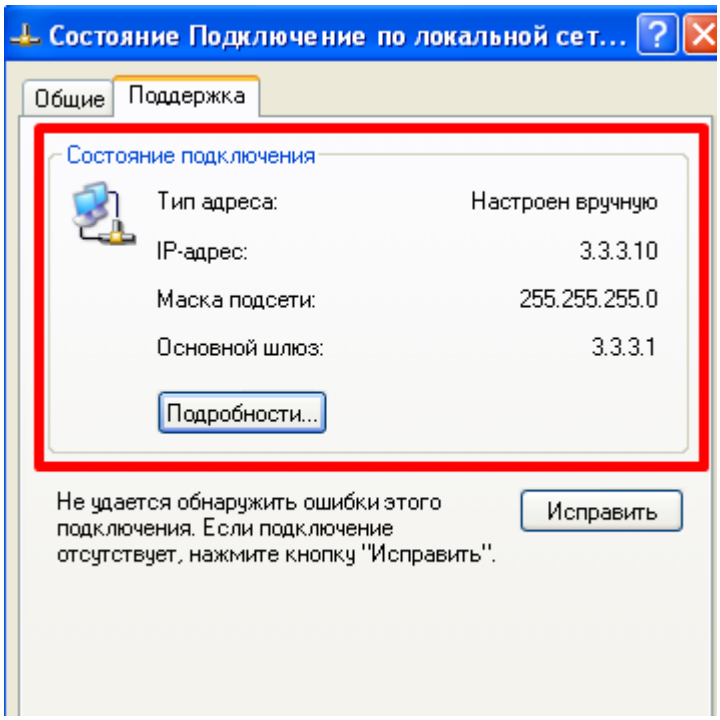

Рисунок 2.2 – Ручне налаштування адреси серверу

Таким самим чином налаштована віртуальна машина, яка виступає у ролі мультикаст клієнта та має адресу 10.10.10.204. Для перевірки з'єднання обох комп'ютерів надішлемо команду ping з комп'ютера 3.3.3.10 до 10.10.10.204 (Рисунок 2.3)

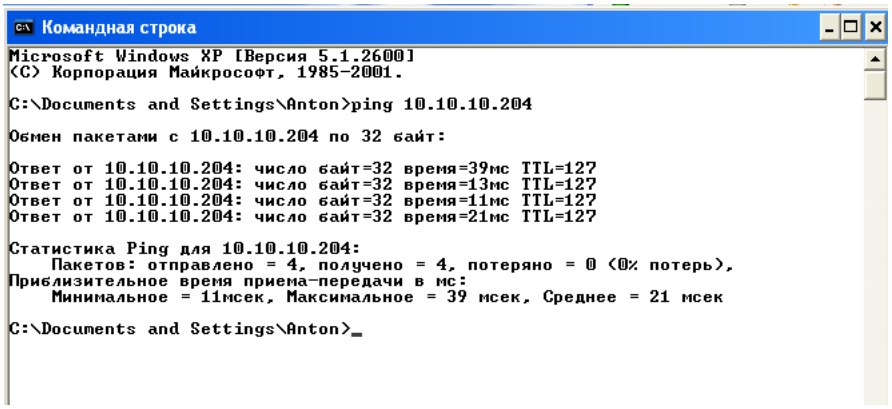

Рисунок 2.3 – Виконання команди ping на одному з комп'ютерів

Після цього можна починати транслювати мультикаст трафік. Запускаємо програму VLC media player на машині сервера. Вмикаємо потік, попередньо вибравши файл для трансляції, протокол RTP (real-time protocol) та вказавши адресу мовлення – 224.2.2.4 і порт 1234. З цього моменту мовлення відео розпочато. Основним моментом є те, що сам відеопотік необхідно приймати не з адреси машини сервера, а вже з адреси мовлення. Адже насправді трафік відеотрансляці надсилається від серверу тільки на адресу мовлення 224.2.2.4, а на інші адреси – ні. Тому щоб отримати його на віртуальній машині-клієнті, запускаємо програму VLC media player, натискаємо «Відкрити медіа-файл» та обираємо трансляцію з мережі, вказавши протокол, адресу мовлення і порт – rtp://224.2.2.4:1234 та натискаємо Відтворити. Відео-потік успішно транслюється з серверу до клієнту (Рисунок 2.4)

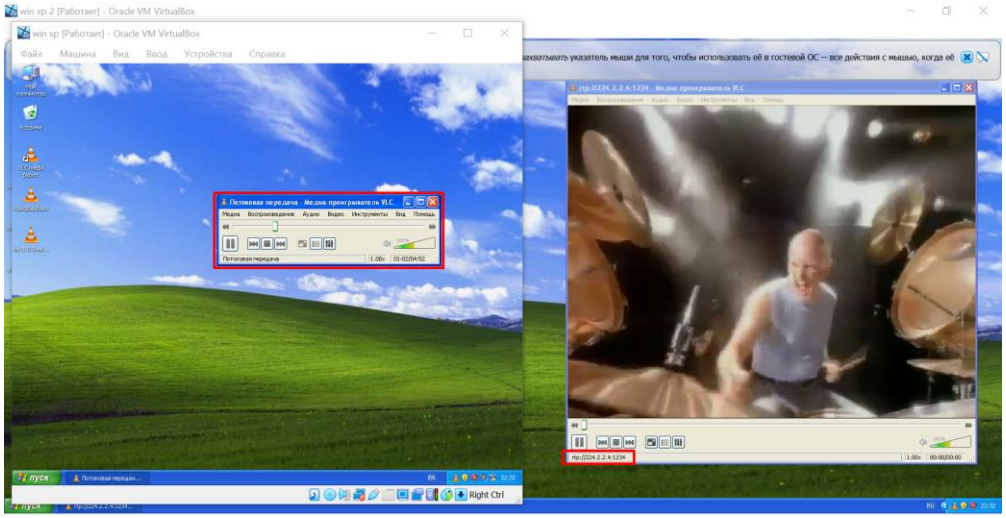

Рисунок 2.4 – Відео-трансляція від машини серверу до машини клієнта

2.3 Налаштування QoS для мережі з IPTV у середовищі GNS3

У змодельованій системі під час трансляції майже увесь трафік складають UDP пакети, які містять данні відео-потоку. Але така ситуація – рідкість, адже у той самий час, користувачі обох машин можуть використовувати мережу і в інших цілях, у мережі може бути більше хостів, роутерів, які спілкуються між собою. Така поведінка є більше наближеною до реальної, а тому відео-транслюванню будуть перешкоджати інші менш важливі пакети трафіку, а це у свою чергу буде значити меншу швидкість скачування відео, та його якість. Щоб трафік IPTV був більш чітким, його необхідно пріоритезувати, для цього нам знадобиться методика QoS, яка допоможе створити спеціальні політики на роутері та відділить потрібний нам трафік від менш важливого. Для цього необхідно скористатися диференційованим обслуговуванням – категорією QoS, що дає змогу відділяти та пріоритезувати трафік за певними правилами. Для налаштування можемо скористуватися уже готовою схемою. Усі конфігурації відбуваються на роутері R1. Для симуляції фонового трафіку скористаємося командою ping і будем надсилати ICMP пакети від одного комп'ютера до іншого. Стандартний розмір пакетів команди ping становить 32 байти і надсилаються у кількості 4 штуки за одну команду. Щоб створити постійний трафік модифікуємо

команду, збільшивши розмір пакету до 1000 байт та зробивши потік безперервним (Рисунок 2.5)

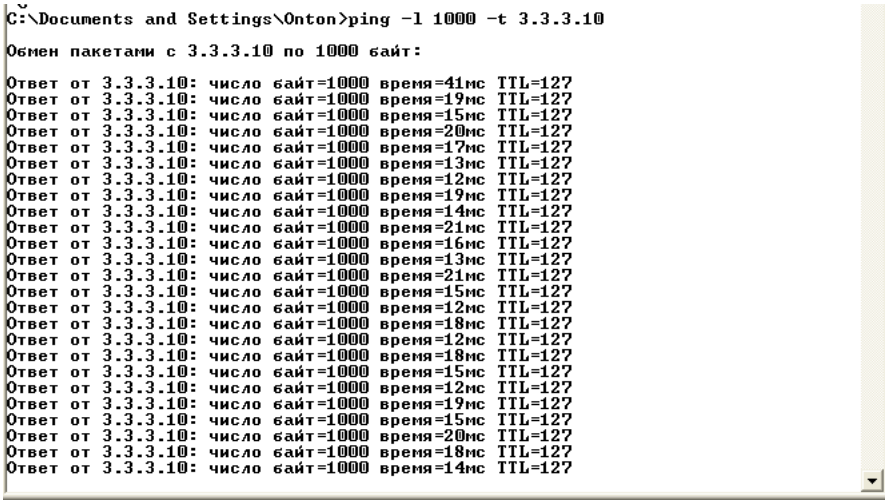

Рисунок 2.5 – Робота модифікованої команди ping

Для того, щоб покращити обслуговування мережі, звернемоя до методології QoS. Спам ICMP трафіку з розміром пакетів у 1000 байт виглядає підозріло і одним з умов налаштування якості обслуговування є фільтрація цього трафіку, а саме блокування пакетів від клієнта до сервера з протоколом ICMP та розміром пакетів більше 500 байтів.

Перейдемо до налаштування роутеру з нашої схеми

R1(config)#access-list 1 permit 10.10.10.0 0.0.0.255 // створення списку доступу

R1(config)#class-map match-all BIG_ICMP // створення class-map

R1(config-cmap)#match access-group 1

R1(config-cmap)#match protocol icmp // вказівка на тип протоколу

R1(config-cmap)#match packet length min 501 // вказівка на розмір пакету

R1(config-cmap)#policy-map ICMP_POLICE // створення policy-map

R1(config-pmap)#class BIG_ICMP

R1(config-pmap-c)#drop // означення того, що буде виконано при збігу з умовами вище

R1(config-pmap-c)#int fa 0/0

R1(config-if)#service-policy output ICMP_POLICE // застосування політики на порті роутера в напрямку серверу

Перевіримо обробку пакетів командою ping з різним розміром (Рисунок 2.6)

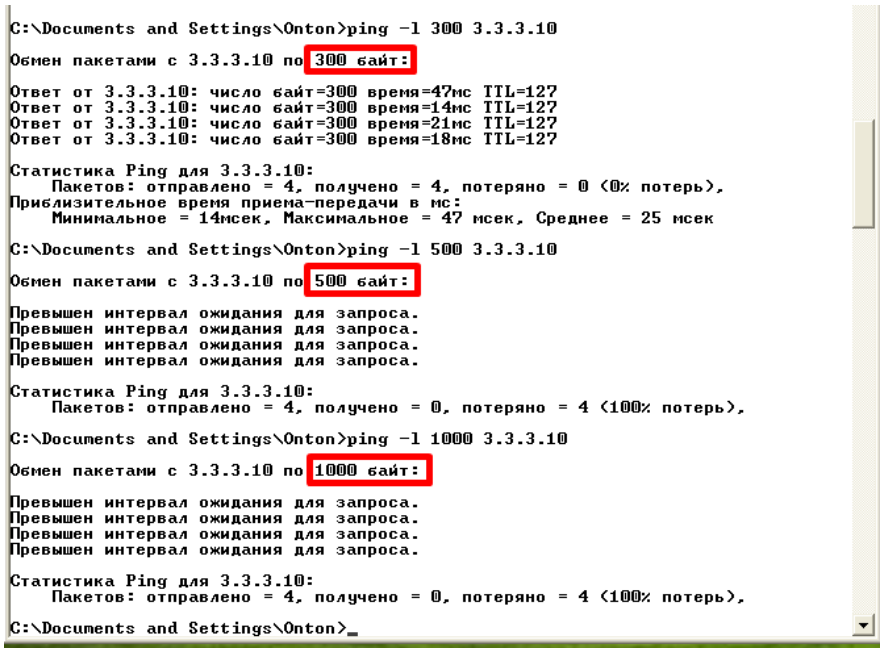

Рисунок 2.6 – Результат команди ping з різним розміром пакетів ICMP

Засмічувати та перешкоджати трафіку IPTV від серверу можуть не лише пакети ICMP, але і інші пакети, наприклад пакети IP-телефонії технології VoIP, у випадку якщо ця технологія буде підключена пізніше, а також інші машини підключені разом із сервером до однієї групи. Тому ще одном способом покращити якість обслуговування даної мережі можна вказавши пріоритет для UDP пакетів надісланих з машини-серверу та виділити для цього трафіку 85 відсотків усього трафіку у мережі. Щоб налаштувати цей пріоритет відкриємо конфігурацію роутеру R1:

R1(config)#access-list 123 permit udp host 3.3.3.10 any // створення нового списку доступу для upd пакетів від машини серверу у будь-якому напрямку

R1(config)#class-map match-all IPTV

R1(config-cmap)#match access-group 123

R1(config-cmap)#policy-map IPTV_TOP

R1(config-pmap)#class IPTV

R1(config-pmap-c)#priority percent 85 // вказівка пріоритету та процентного відношення трафіку

R1(config-pmap-c)#int fa0/1

R1(config-if)#service-policy output IPTV_TOP // вказівка на порт роутеру та напрямок трафіку

2.4 Аналіз QoS для мережі з IPTV за допомогою аналізатора WireShark

Після того, як мережа налаштована, необхідно перевірити її на наявність помилок, проаналізувати трафік аби виявити слабкі місця або впевнетися у правильності налаштування мережі. Щоб проаналізувати трафік звернемося до програми-сніферу WireShark. У останніх версіях програми GNS3 цей аналізатор вже інтегрований до середовища. Якщо використовується більш рання версія, то аналізатор можна скачати окремо та інтегрувати у меня налаштувань GNS3. Програма-сніфер перехоплює трафік лише з ліній зв'язку, а не на самих маршрутизаторах чи машинах, тому щоб почати аналіз, необхідно натиснути правою кнопкою миші на ланку зв'язку, яку необхідно проаналізувати та натиснути «Почати перехоплення». Після цього відбувається запуск програми WireShark і одразу можна спостерігати за усим трафіком онлайн (Рисунок 2.7)

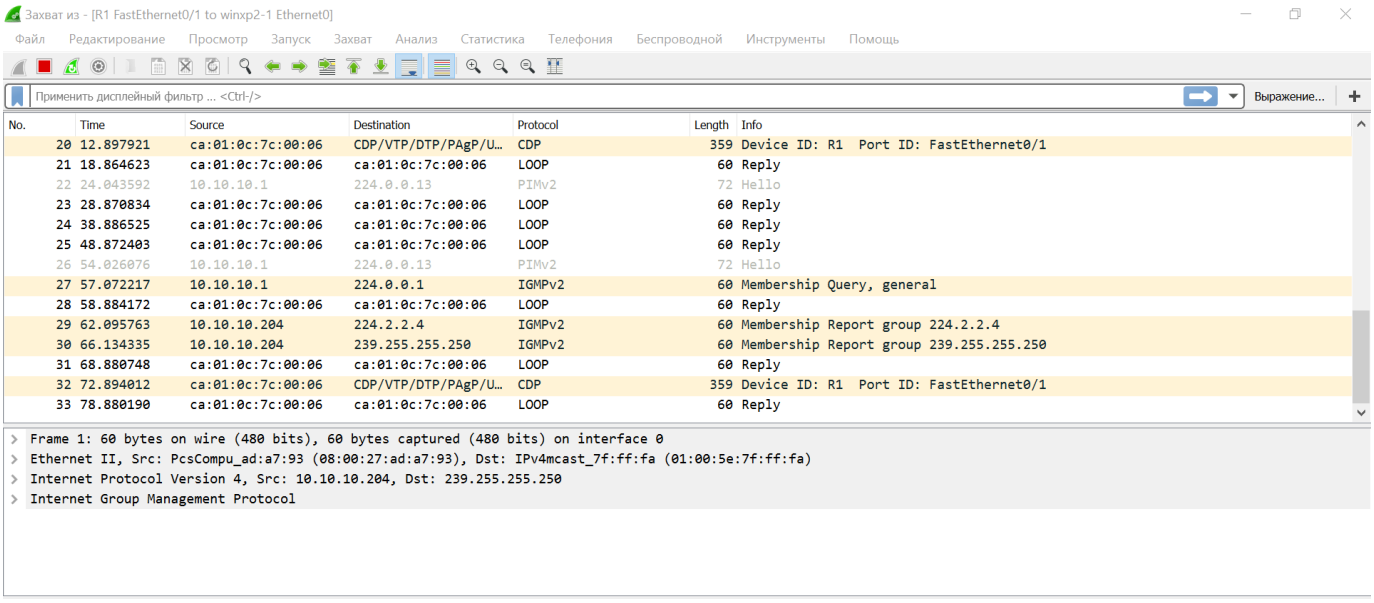

Рисунок 2.7 – Перехоплення трафіку між машиною-клієнтом та роутером R1

Проаналізувавши трафік одразу видно поведінку роботи технології мультикаст: раз у деякий проміжок часу від роутера до клієнта надходить запит про членство – Membership Query по протоколу IGMPv2, а у відоповідь клієнт відповідає звітом про членство – Membership Report, що він знаходиться у певній групі та готовий отримувати мультикастовий трафік. Для аналізу інших пакетів необхідно їх симулювати. Скористаємося командою ping та надешлемо декілька повідомлень від клієнта до серверу (Рисунок 2.8)

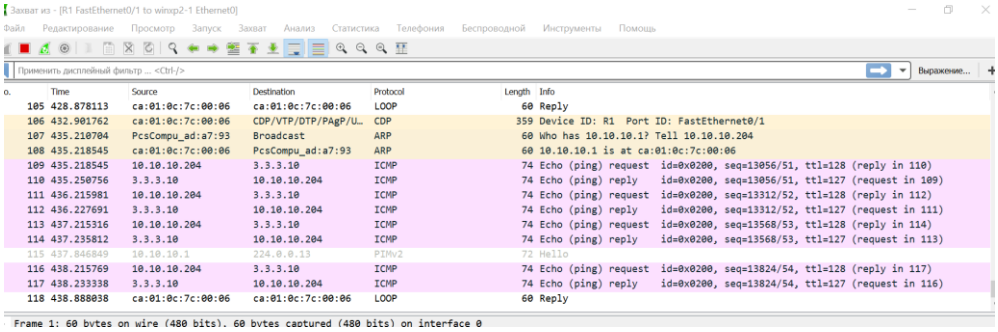

Рисунок 2.8 – Перехоплення трафіку команди ping

З рисунку видно, що з машини 10.10.10.204 до машини 3.3.3.10 надійшло 4 пакети протоколу ICMP по 74 байти, та 4 пакети у відповідь. Ці пакети підсвічені програмою WireShark рожевим для зручності. На Рисунку 2.9 зображено обмін повідомленнями за допомогою команди net send

No.	Time	Source	Destination	Protocol	Length	Info
200	708.872354	ca:01:0c:7c:00:06	ca:01:0c:7c:00:06	LOOP	60	Reply
201	717.077285	10.10.10.1	224.0.0.1	IGMPv2	60	Membership Query, general
202	718.877948	ca:01:0c:7c:00:06	ca:01:0c:7c:00:06	LOOP	60	Reply
203	721.931013	10.10.10.204	224.2.2.4	IGMPv2	60	Membership Report group 224.2.2.4
204	721.931013	10.10.10.204	239.255.255.250	IGMPv2	60	Membership Report group 239.255.255.250
205	727.179185	10.10.10.204	3.3.3.10	NBNS	92	Name query NBSTAT *<00><00><00><00><00><00><00><00><00><00><00><00><00><00><…
206	727.199682	3.3.3.10	10.10.10.204	NBNS	289	Name query response NBSTAT
207	727.199682	10.10.10.204	3.3.3.10	TCP	62	1037 → 139 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
208	727.221153	3.3.3.10	10.10.10.204	TCP	62	139 → 1037 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1
209	727.221153	10.10.10.204	3.3.3.10	NBSS	126	Session request, to ANTON-PC<03> from ONTON-PC<03>
210	727.242626	3.3.3.10	10.10.10.204	NBSS	58	Positive session response
211	727.242626	10.10.10.204	3.3.3.10	SMB	125	Send Single Block Message Request
212	728.057290	3.3.3.10	10.10.10.204	TCP	54	139 → 1037 [ACK] Seq=5 Ack=144 Win=64097 Len=0
213	728.861203	ca:01:0c:7c:00:06	ca:01:0c:7c:00:06	LOOP	60	Reply
214	729.086193	3.3.3.10	10.10.10.204	SMB	93	Send Single Block Message Request
215	729.086193	10.10.10.204	3.3.3.10	TCP	60	1037 → 139 [FIN, ACK] Seq=144 Ack=44 Win=64197 Len=0
216	729.107662	3.3.3.10	10.10.10.204	TCP	54	139 → 1037 [FIN, ACK] Seq=44 Ack=145 Win=64097 Len=0
217	729.107662	10.10.10.204	3.3.3.10	TCP	60	1037 → 139 [ACK] Seq=145 Ack=45 Win=64197 Len=0
218	732.902176	ca:01:0c:7c:00:06	CDP/VTP/DTP/PAgP/U...	CDP	359	Device ID: R1 Port ID: FastEthernet0/1
219	734.648454	10.10.10.1	224.0.0.13	PIMv2	72	Hello
220	738.870716	ca:01:0c:7c:00:06	ca:01:0c:7c:00:06	LOOP	60	Reply

Рисунок 2.9 – Перехоплення пакетів команди net send

Можна спостерігати пакети TCP протоколу. Можливості програми сніфера дозволяють відстежити, що саме було надіслано, з якого комп'ютера надіслано та до якої машини призначено. Щоб відстежити повідомлення, натиснемо правою кнопкою на TCP пакет та виберемо «Відстежувати трафік TCP». У результаты програма-аналізатор збере усю необхідну інформацію про обмін повідомленням та покаже на екрані (Рисунок 2.10)

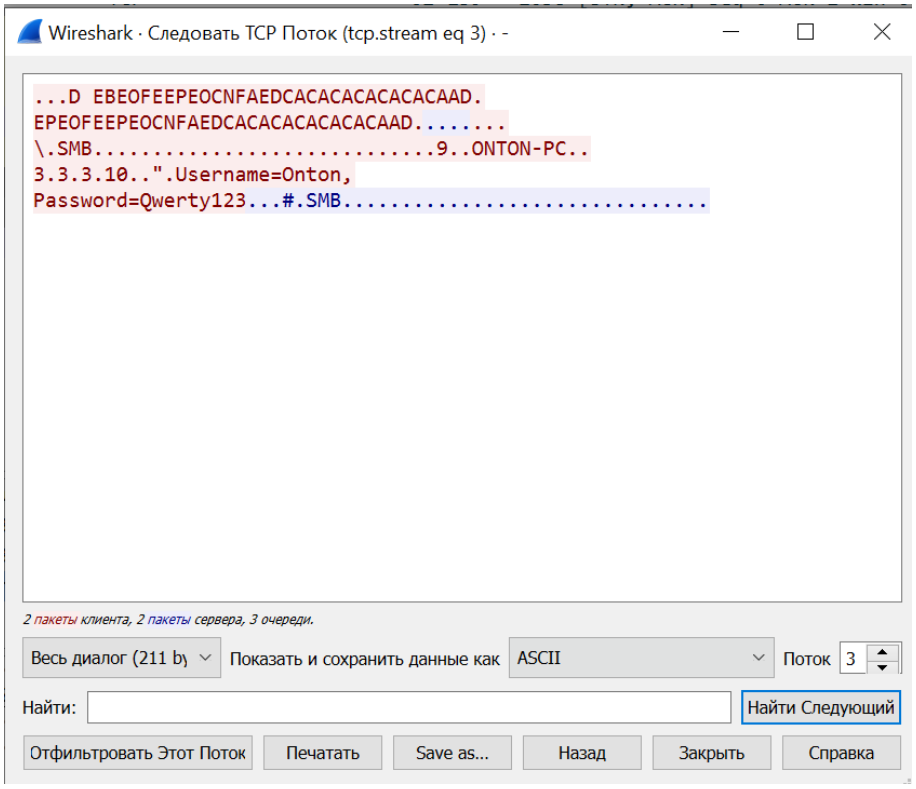

Рисунок 2.10 – Перехоплення повідомлення надісланого з машини

Перехоплений потік пакетів показує, що повідомлення було надіслано від користучава з ніком Onton-PC та надіслано до адреси 3.3.3.10, що є машиною сервера, також можна спостерігати текст повідомлення, який може містити будь-яку корисну інформацію. У даному випадку було надіслано текст «Username=Onton, Password=Qwerty123». Також перехоплений потік містить і іншу інформацію про розмір повідомлення, час відправки та інше.

Симулюємо навантаження мережі за допомогою команди ping та почнемо трансляцію відео контенту з віртуальної машини серверу, отримуватиме відео потік віртуальна машина-клієнт.

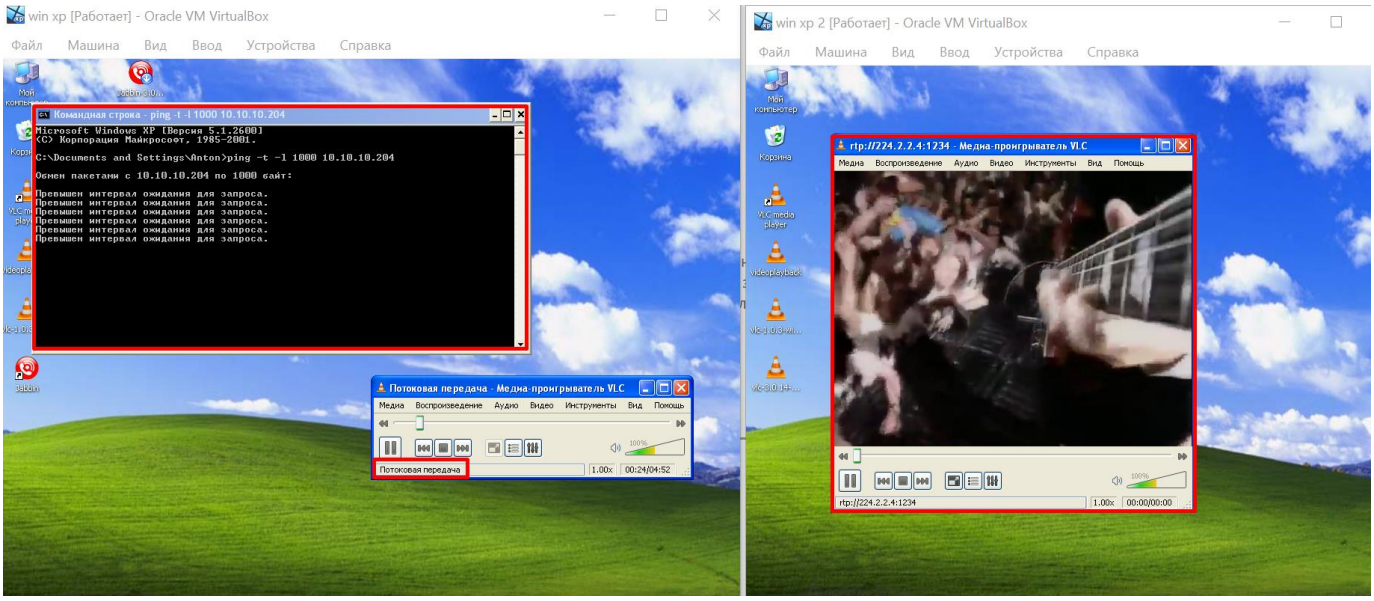

Рисунок 2.11 – Трансляція IPTV під час навантаження пакетами по 1000 байтів

Проаналізувавши трафік за допомогою програми WireShark можна побачити, що пакети трафіку від команди ping не надсилаються від машини до машини, а це у свою чергу означає, що нічого не перешкоджає пріоритезованому трафіку багатоадресного мовлення IPTV та надсиланню пакетів протоколу UDP до выртуальної машини клієнта. Таким чином переконалися у справному налаштуванні мережі та вдалому використання методології QoS (Рисунок 2.12)

No.	Time	Source	Destination	Protocol	Length	Info
4790	2253.572158	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4791	2253.582896	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4792	2253.593632	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4793	2253.604366	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4794	2253.615101	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4795	2253.625838	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4796	2253.636574	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4797	2253.647312	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4798	2253.658048	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4799	2253.658048	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4800	2253.668783	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4801	2253.676592	10.10.10.204	3.3.3.10	ICMP	1042	Echo (ping) request id=0x0200, seq=17408/68, ttl=128 (no response found!)
4802	2253.679519	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4803	2253.690256	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4804	2253.700992	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4805	2253.711727	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328
4806	2253.722470	3.3.3.10	224.2.2.4	UDP	1370	1040 → 1234 Len=1328

Рисунок 2.12 – Перехоплення IPTV трафіку та команди ping

2.5 Реалізація графічного інтерфейсу налаштування мережі мовою JavaScript

Для реалізації графічного інтерфейсу у даній роботі використовується об’єктно-орієнтовна мовна програмування JavaScript. Ця мова була обрана через свою простоту в реалізації, адже скрипти – програми, створені цією мовою, не потребують додаткових двигунів чи програмного забезпечення для запуску. Скрипт можна записати у звичайний html файл, та запустити за допомогою браузера. Крім браузерів скрипти мовою JavaScript на сьогоднішній день також можна запускати не лише у браузерах, а і на серверах або навіть будь-яких пристроях, на яких встановлений спеціальний двигун JavaScript engine. Також JavaScript є безпечною в плані того, що вона не дає прямого доступу до заліза комп'ютера (пам'яті чи процесора), вона виступає скоріше як «оживлювач» веб-сторінки. Використовуючи браузер, як середовище розгортання JavaScript програм, скрипти роблять усе, що можна пов'язати з маніпуляціями та модифікаціями наповнення веб-сторінки, а саме: створення html-сторінок, модифікація стилів, оформлення, тексту та наповнення усієї сторінки в один клік мишки.

Також ця мова програмування має багато переваг перед іншими популярними мовами. Синтаксис вважається С-подібним, але JavaScript навідміну від мов С являється динамічним, що означає, що змінні під час виконання коду програми можуть змінювати свій тип, коли у мові С++ змінні чітко статичні. JS більше схожий на мову Java, але значно спрощенний порівняно з ним. Для реалізації задачі даної роботи JavaScript відповідає усім вимогам.

3 СТВОРЕННЯ СХЕМИ IPTV З ВИКОРИСТАННЯМ МЕТОДОЛОГІЇ QOS У ПРОГРАМНОМУ СЕРЕДОВИЩІ

3.1 Розробка графічного інтерфейсу налаштування QoS для мережі з IPTV

У середовищі GNS3 було змодельовано локальну мережу, де було налаштовано маршрутизатори, зконфігуровано технологію багатоадресного мовлення multicast для реалізації IPTV за допомогою протоколів IGMP та PIM, покращено якість обслуговування QoS та проаналізовано усю мережу за допомогою аналізатора WireShark. За результатами роботи, що була проведена, стало зрозуміло, що багато часу було витрачено на налаштування роутерів, а в середовищі GNS3 відсутній графічний інтерфейс для прискорення рутинної конфігурації маршрутизаторів. Тому актуальною задачею є розробка інтерфейсу для автоматизації налаштування multicast мовлення трафіку та налаштування методології QoS для такої мережі.

Розробка проекту відбувалася за допомогою мови програмування JavaScript та розмітки html та стилів CSS.

Інтерфейс представляє собою розміщену по центру схему з трьох роутерів та двох машин, які виступають у ролі сервера та клієнта. Біля кожного з роутерів знаходяться поля для заповнення необхідних даних, а саме: IP номер порту та маска. Також є окреме поле для введеня IP адреси машини серверу. Усі ці данні необхідні під час генерації скрипту для налаштування маршрутизаторів. Після того, як користувач заповнить усі необхідні поля, необхідно натиснути кнопку “Generate Settings”. Після цього унизу буде згенеровано усі необхідні скрипти, які можна легко зкопіювати для подальшої роботи зі справжніми маршуртизаторами. На Рисунку 3.1 та Рисунку 3.2 нижче зображено графічний інтерфейс сторінки.

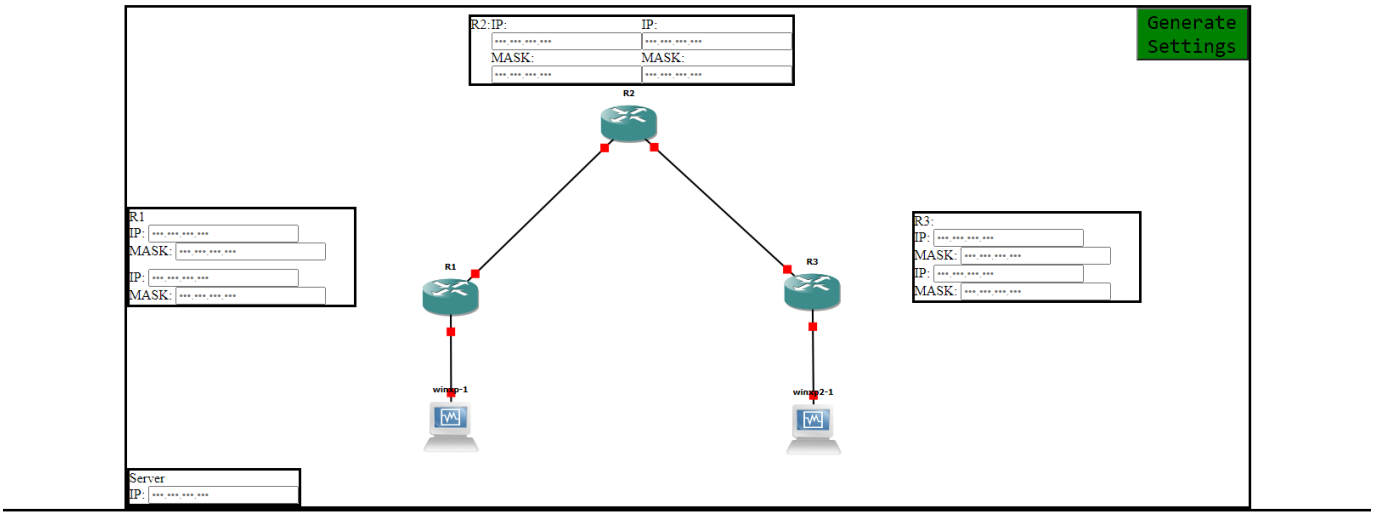

Рисунок 3.1 – Графічний інтерфейс програми для введення полів

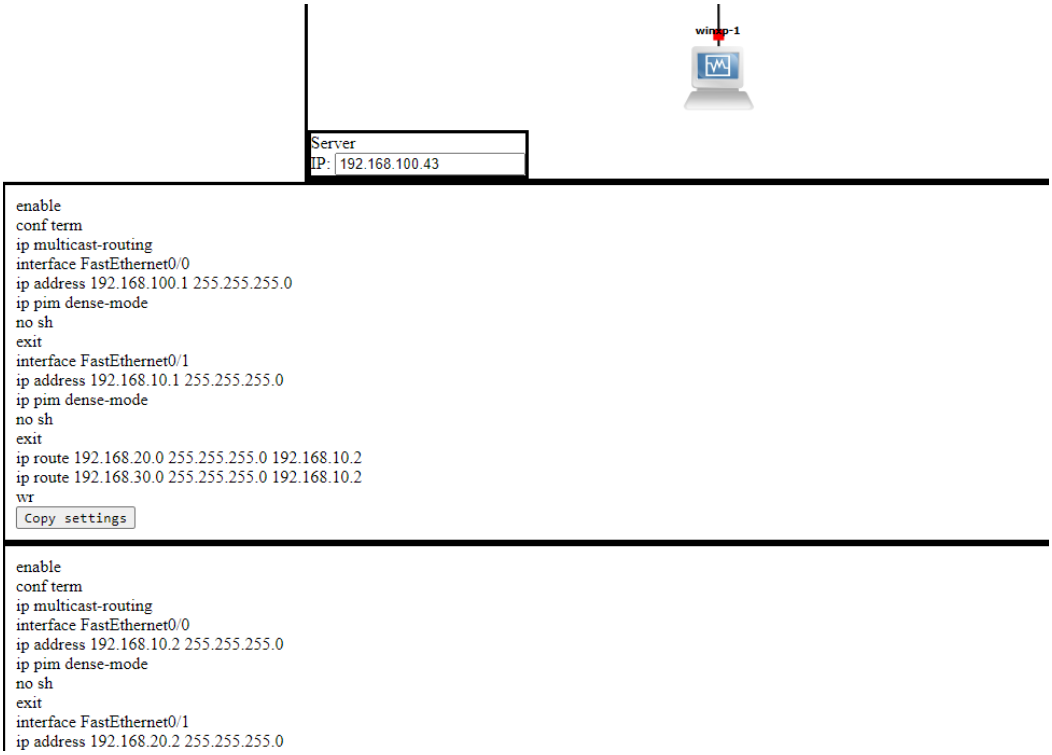

Рисунок 3.2 – Графічний інтерфейс згенерованих скриптів

Після того, як скрипти для конфігураціх були згенеровані, користувач може приступати до налаштування маршуртизаторів та інших машин

3.2 Тестування створеного інтерфейсу в емуляторі GNS3

Працездатність даної системи зпочатку необхідно протестувати. Для цього відкриємо графічний інтерфейс та заповнимо усі необхідні поля. Після заповнення

необхідно натиснути кнопку “Generate Settings” та знавігуватися нижче для того, щоб скопіювти згенеровані скрипти (Рисунок 3.3).

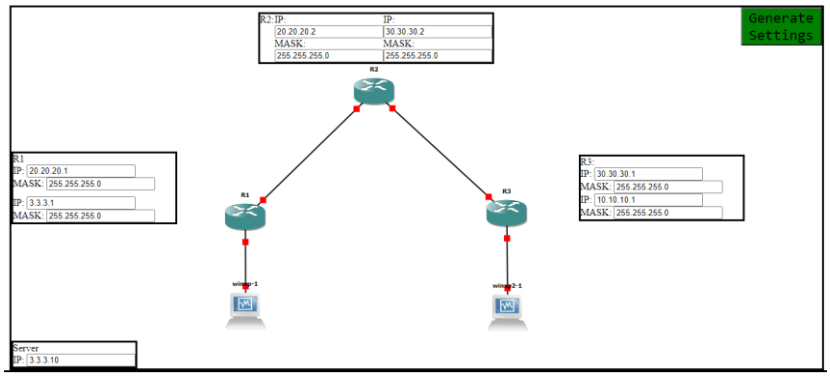

Рисунок 3.3 – Заповнення полів у системі

Наступним кроком є відтворення даної схеми у емуляторі GNS3. Схема виглядає наступним чином (Рисунок 3.4)

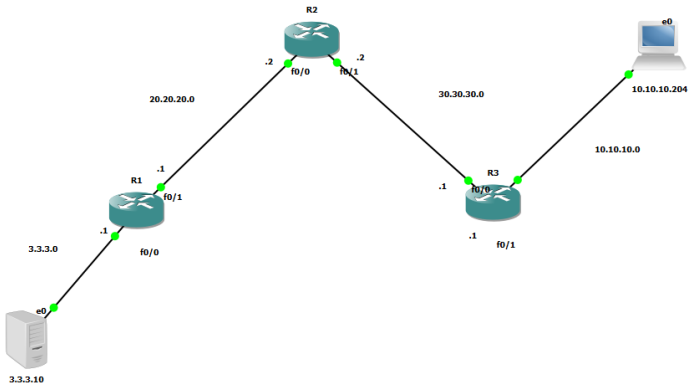

Рисунок 3.4 – Схема в емуляторі GNS3

Переходимо до налаштування роутерів. Маючи готові конфігураційні скрипти, їх достаньо просто скопіювати та вставити до відповідних роутерів. Налаштуємо перший роутер згідно згенерованого коду у графічному інтерфейсі. На Рисунку 3.5 зображено, як саме виглядає скопійований скрипт у самій консолі налаштування роутеру

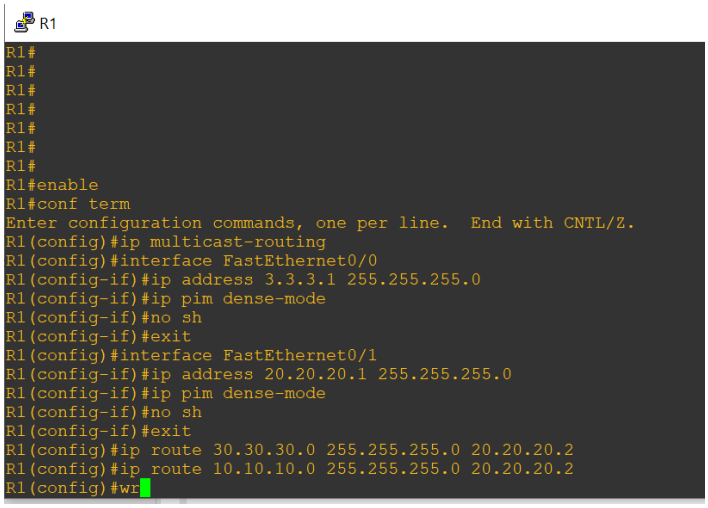

Рисунок 3.5 – Перенесений скрипт зі створеної програми до консолі налаштування роутера R1

За аналогією налаштовуються і інші роутери. Після закінчення налаштувань також необхідно налаштувати IP адреси підключених машин. Запускаэмо трансляцію відео-потоку з машини сервера та приймаємо потік на машині-клієнті. На Рисунку 3.6 добре видно, що IPTV налаштовано добре і потік стабільний

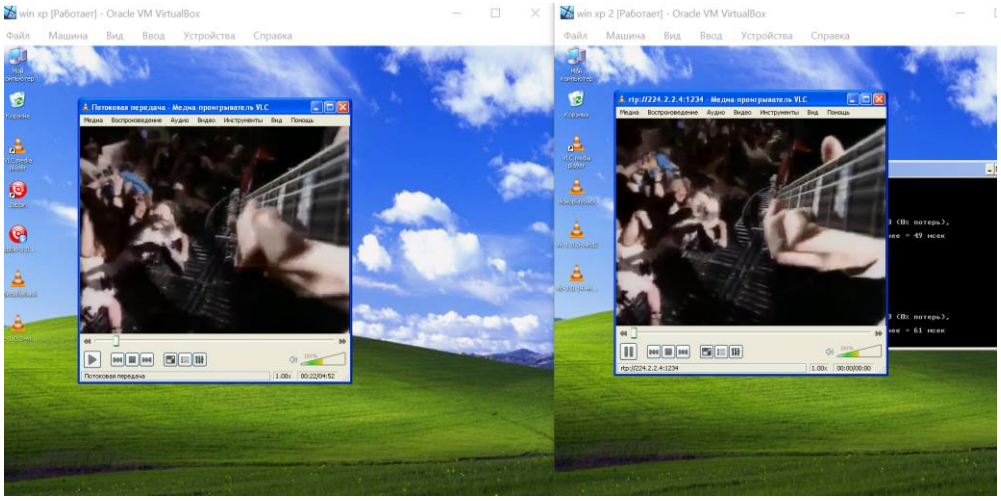

Рисунок 3.6 – IPTV трансляція з машини серверу до машини клієнта базуючись на налаштуваннях згенерованих у створенійінформаційній системі

Також для перевірки налаштування даної мережі можна скористуватися аналізатором програмою-сніфером WireShark. Для цього необхідно здійснити підключення до однієї з ланок зв’язку між роутерами чи машинами та почати

перехоплення. Під час перехоплення трафіку явно видно пакети протоколу UDP, що надсилаються від машини серверу до широкомовної адреси 224.2.2.4 з портом 1234. Також крім пакетів відео-потоку на каналі відстежуються пакети протоколу IGMPv2 та PIMv2, що відповідають за багатоадресне мовлення. Крім аналізу за допомогою аналізатора WireShark також можна перевірити трафік на роутері за допомогою команди show ip mroute D (прапор D вказує на dense-mode), яка також використовується для аналізу мультикастового трафіку. З її допомогою видно трансляцію з адреси 3.3.3.10 на 224.2.2.4 (Рисунок 3.7)

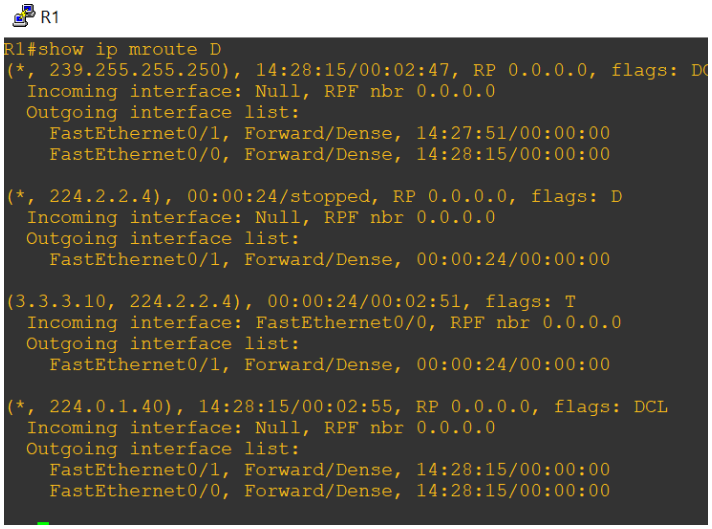

Рисунок 3.7 – Аналіз налаштувань схеми за допомогою команди show ip mroute

За допомогою двох віртуальних машин, які виступали у ролі машини-серверу та машини-клієнта та роутерів Cisco 7200 було створено та налаштовано комп’ютерну мережу. Було сконфігуровано технологію передачі multicast трафіку від серверу до користувача-клієнта, а також покращено якість обслуговування даної мережі за методологією QoS. На такий тест витрачається значно менше часу, ніж при повній конфігурації маршрутизаторів з нуля. Адже під час налаштувань вручну багато часу відходить на рутину конфігурацію кожного з портів. Протестувавши розроблену систему можна стверджувати, що вона значно покращує, спрощує та пришвидшує налаштування технології IPTV та методологію QoS для такої мережі.

ВИСНОВКИ

У ході даної роботи було розглянуто принципи роботи основних інструментів системного адміністрування при налаштуванні комп'ютерних мереж, а саме емулятор з графічним інтерфейсом GNS3, аналізатор мережевого трафіку WireShark. Також було розглянуто та досліджено технологію багатомовного транслювання медіа-файлів IPTV та відповідні технології протоколи IGMP та PIM, за допомогою яких таку трансляцію було реалізовано. Також досліджено основні проблеми при трансляції IPTV та способи покращення якості обслуговування QoS комп'ютерної мережі. Було змодельовано комп'ютерну мережу з технологією IPTV, покращено якість обслуговування такої мережі та глибоко проаналізовано трафік у такій мережі під час звичайної трансляції та під час навантажень іншим трафіком.

У ході роботи було виявлено, що основним недоліком сучасних симуляторів та емуляторів комп'ютерних мереж є те, що налаштування та конфігурування маршрутизаторів для таких мереж є довготривалим процесом, адже на рутинне налаштування роутерів відходить багато часу. Тому для вирішення даної проблеми було розроблено веб-орієнтовну інформаційну систему для генерації скриптів для налаштування маршрутизаторів. У даній системі є можливість ввести лише вхідні дані, що включають у себе IP-адреси інтерфейсів роутеру, IP-вдресу серверу та маски підмереж. А після цього програма генерує потрібні скрипти для відповідних роутерів, що дає можливість легко скопіювати їх та вставити у консоль маршрутизатора.

Отриманий графічний інтерфейс дозволяє налаштовувати транслювання IPTV та застосування методик QoS недосвідченим користувачам навіть без знань базових та основних команд налаштування маршрутизаторів Cisco і тим самим значно прискорюючи процес розгортання та старту комп'ютерної мережі, виключаючи можливі людські помилки та позбавляючи користувача рутинного ручного налаштування.

СПИСОК ЛІТЕРАТУРИ

1. «What is Internet protocol Television IPTV?» [Електронний ресурс] - https://www.nevron.eu/blog/iptv/
2. VOD – видео по заапросу [Електронний ресурс] - http://iptv-mag.ru/VOD.html
3. Комп'ютерні мережі. Протокол IGMP [Електронний ресурс] - http://iptcp.net/protokol-igmp.html
4. Knowledge Center. What is Protocol Independent Multicast (PIM)? [Електронний ресурс] - https://www.metaswitch.com/knowledge-center/reference/what-is-protocol-independent-multicast-pim
5. Multicast PIM Sparse Mode [Електронний ресурс] - https://networklessons.com/multicast/multicast-pim-sparse-mode
6. Шрінівас Вегешна, IP Quality of Service 368, Вільямс 2017 – 368 с.
7. Quality of Service in Networking [Електронний ресурс] - https://www.fortinet.com/resources/cyberglossary/qos-quality-of-service
8. Ukrainian Journal of Educational Studies and Information Technology Vol. 3, No 1 (2016)
9. Network Monitoring. SolarWinds vs Wireshark [Електронний ресурс] - https://stackshare.io/stackups/solarwinds-npm-vs-wireshark
10. Офіційний сайт продукту WireSkark [Електронний ресурс] - https://www.wireshark.org/
11. Сетевые симуляторы и эмулятори оборудования Cisco [Електронний ресурс] - https://top-technologies.ru/ru/article/view?id=38134
12. Бодчер Р. Программа сетевой академии Cisco CCNA [3-е изд.] : [пер. с англ.] Рональд Бодчер, К. Р. Киркендаль. – М. : изд. Дом "Вильямс", 2005. – 1186 с.
13. IP Multicast Routing Configuration Guide, Cisco, p. 17-19, retrieved 2017-05-27
14. Кріс Сандерс Анализ пакетов. Практичесое руководство по использованию WireShark и tcpdump 2016. – 448 с.

15. КАК ПОЛЬЗОВАТЬСЯ WIRESHARK ДЛЯ АНАЛИЗА ТРАФИКА [Електронний ресурс] // losst.ru. – 2016. – Режим доступу до ресурсу: https://losst.ru/kak-polzovatsya-wireshark-dlya-analiza-trafika.
16. ACL: списки контроля доступа в Cisco IOS [Електронний ресурс] - https://habr.com/ru/post/121806/
17. Протал документации S-Terra. Документация команды class-map [Електронний ресурс] - https://doc.s-terra.ru/rh_output/4.2/Gate/output/index.htm#t=mergedProjects%2FConsole%2Fclass-map.htm
18. NetworkGuru. Wireshark фильтр по IP, по порту, по протоколу, по MAC [Електронний ресурс] - https://networkguru.ru/wireshark-filtr-po-ip-portu-protokolu-mac/
19. About multicast routing. Help center [Електронний ресурс] - https://www.watchguard.com/help/docs/help-center/en-US/Content/en-US/Fireware/networksetup/multicast_about_c.html
20. DISCOVER THE LIST OF 45 JAVASCRIPT COMMANDS IN 2021 [Електронний ресурс] – https://bytescout.com/blog/javascript-commands.html
21. About JavaScript. What is JS? [Електронний ресурс] - https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript

ДОДАТОК

Додаток А

Page.html

```
<!DOCTYPE html>
<html>
   <head>
      <title>Titariev Anton</title>
      <meta charset="utf-8">
      <style type="text/css">
         .backgrnd {
         margin: 0 auto;
         width: 70%;
         display: table;
         border: solid;
         }
         .left {
         display: table-cell;
         width: 20%;
         position: relative;
         }
         .center {
         display: table-cell;
         width:50%;
         }
         .right {
         display: table-cell;
         position: relative;
```

```
width: 20%;
}
.router1 {
border: solid;
position: absolute;
top: 50%;
transform: translate(0, -50%);
width: 100%;
}
.router2 {
border: solid;
margin: auto;
width: 50%;
display: table
}
.router3 {
border: solid;
position: absolute;
top: 50%;
transform: translate(0, -50%);
width: 100%;
}
.r1-1{
margin-bottom: 10px
}
.r1-2{

}
```

```
.r2-1{
display: table-cell;
}
.r2-2{
display: table-cell;
}
.r3-2{

}
.r3-2{

}
.center-bot {
text-align:center;
}
.textarea-1 {
background: #000 none repeat scroll 0 0;
}
.server {
border: solid;
position: absolute;
bottom: 0;
}
.set-r1{
border:solid;
padding: 10px;
}
.set-r2{
```

```
            border:solid;
            padding: 10px;
            }
            .set-r3{
            border:solid;
            padding: 10px;
            }
            .generate-btn{
            background-color: green;
            }
        </style>

    </head>
    <body>
        <div class='backgrnd'>
        <div class='left'>
     <div class='router1'>
     R1
            <div class='r1-1'>
            IP: <input type="text" id="Input3" title="Write ip"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
                                <br
/>
MASK: <input type="text" id="Input4" title="Write mask"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
            </div>
```

```
        <div class='r1-2'>
        IP: <input type="text" id="Input1" title="Write ip"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
                                <br
/>
MASK: <input type="text" id="Input2" title="Write mask"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
        </div>
    </div>
    <div class='server'>
    Server </br>
        IP: <input type="text" id="InputServ" title="Write
ip"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
    </div>
</div>
<div class='center'>
    <div class='center-top'>
        <div class='router2'>
        R2:
            <div class='r2-1'>
            IP: <input type="text" id="Input5" title="Write
ip"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
```

```
                                <br
/>
MASK: <input type="text" id="Input6" title="Write mask"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
            </div>
            <div class='r2-2'>
            IP: <input type="text" id="Input7" title="Write
ip"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
                                <br
/>
MASK: <input type="text" id="Input8" title="Write mask"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
            </div>
        </div>
    </div>
    <div class='center-bot'>
        <img src="ip schema.png">
    </div>
</div>
<div class='right'>
    <div class='router3'>
    R3:
        <div class='r3-1'>
        IP: <input type="text" id="Input9" title="Write ip"
```

```
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
                                <br
/>
MASK: <input type="text" id="Input10" title="Write mask"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
        </div>
        <div class='r3-2'>
        IP: <input type="text" id="Input11" title="Write ip"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
                                <br
/>
MASK: <input type="text" id="Input12" title="Write mask"
placeholder='•••.•••.•••.•••'
pattern="\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}">
        </div>
    </div>
</div>
<button onclick="myFunction()" class="generate-btn"
style="margin: 0
auto;font-size:24px;font-family:monospace">Generate
Settings</button>

</div>
<div class='bot'>
    <div class='set-r1'>
```

```
        <div id="textarea-1">enable<br />
conf term<br />
ip multicast-routing <br />
interface FastEthernet0/0 <br />
ip address <span id="r11ip"/></span> <span
id="r11mask"></span> <br />
ip pim dense-mode <br />
no sh <br />
exit <br />
interface FastEthernet0/1<br />
ip address <span id="r12ip"></span> <span
id="r12mask"></span><br />
ip pim dense-mode <br />
no sh <br />
exit<br />
ip route <span id="r2r3net"></span> <span
id="r31mask"></span> <span id="r21ip"></span><br/>
ip route <span id="r3clientnet"></span> <span
id="r32mask"></span> <span id="r21ip2"></span><br/>
wr</div>
<button class="btn-clipboard2"
onclick="CopyToClipboard1('textarea-1')"
data-clipboard-target="#textarea-1" style="margin: 0
auto;fontsize:18px;font-family:monospace">Copy
settings</button>
    </div>
    <div class='set-r2'>
        <div id="textarea-2">enable<br />
```

```
conf term<br />
ip multicast-routing <br />
interface FastEthernet0/0<br />
ip address <span id="r21ip3"></span> <span
id="r21mask"></span><br />
ip pim dense-mode <br />
no sh <br />
exit <br />
interface FastEthernet0/1<br />
ip address <span id="r22ip"></span> <span
id="r22mask"></span><br />
ip pim dense-mode <br />
no sh <br />
exit<br />
ip route <span id="ipserv1"></span> <span
id="r11mask3"></span> <span id="r12ip2"></span><br/>
ip route <span id="ip3client"></span> <span
id="r32mask2"></span> <span id="r31ip"></span><br/>
wr<br />

conf term<br />
access-list 123 permit udp host <span
id="servip"></span>  any any<br />
class-map match-all IPTV<br />
match access-group 123<br />
policy-map IPTV_POL<br />
class IPTV<br />
priority percent 90<br />
```

```
int fa 0/1<br />
service-policy output IPTV_POL<br />
</div>
<button class="btn-clipboard2"
onclick="CopyToClipboard1('textarea-2')"
data-clipboard-target="#textarea-2" style="margin: 0
auto;fontsize:18px;font-family:monospace">Copy
settings</button>
    </div>
    <div class='set-r3'>
        <div id="textarea-3">enable<br />
conf term<br />
ip multicast-routing <br />
interface FastEthernet0/0<br />
ip address <span id="r31ip2"></span> <span
id="r31mask2"></span><br />
ip pim dense-mode <br />
no sh <br />
exit <br />
interface FastEthernet0/1<br />
ip address <span id="r32ip"></span> <span
id="r32mask3"></span><br />
ip pim dense-mode <br />
no sh <br />
exit<br />
ip route <span id="servr1net"></span> <span
id="r11mask2"></span> <span id="r22ip2"></span><br/>
```

```
ip route <span id="r1r2net"></span> <span
id="r12mask"></span> <span id="r22ip3"></span><br/>
wr <br />
conf term <br />
access-list 1 permit <span
id="ip3clientnet2"></span> <span id="rev32mask"></span>
</br>
class-map match-all BIG_ICMP </br>
match access-group 1  </br>
match protocol icmp   </br>
match packet length min 500    </br>
policy-map ICMP_POLICE    </br>
class BIG_ICMP     </br>
drop   </br>
int fa 0/0      </br>
service-police output ICMP_POLICE      </br>
</div>
<button class="btn-clipboard2"
onclick="CopyToClipboard1('textarea-3')"
data-clipboard-target="#textarea-3" style="margin: 0
auto;fontsize:18px;font-family:monospace">Copy
settings</button>
    </div>
</div>
   </body>
   <script src="http://code.jquery.com/jquery-1.11.0.min.js"
type="text/javascript"/>
```

```
      <script
src="https://cdn.rawgit.com/zenorocha/clipboard.js/master/dis
t/clipboard.min.js"></script>
   <script type="text/javascript">
      function myFunction() {
 var a = document.getElementById("Input1").value; // r1-1 ip
 var aa = a;
 var b = document.getElementById("Input2").value; // r1-1
mask
var bb =b;
var bbb =bb;
 var c = document.getElementById("Input3").value; // r1-2 ip

 var d = document.getElementById("Input4").value; // r1-2
mask
var dd = d;
 var e = document.getElementById("Input5").value; // r2-1 ip
var ee = e;
var eee=ee;
 var f = document.getElementById("Input6").value; // r2-1
mask

 var g = document.getElementById("Input7").value; // r2-2 ip
var gg = g;
var ggg = gg;
 var h = document.getElementById("Input8").value; // r2-2
mask
```

```
 var j = document.getElementById("Input9").value; // r3-1 ip
 var jj = j;
 var k = document.getElementById("Input10").value; // r3-1
mask
 var kk = k;
 var l = document.getElementById("Input11").value; // r3-2 ip

 var m = document.getElementById("Input12").value; // r3-2
mask
 var mm = m;
 var mmm = mm;
 var serv = document.getElementById("InputServ").value;
//server IP

var ipserv1 = (a.trim()).replace(/\d{1,3}$/,"0"); //network
ip between server and R1
var ip12 = (dd.trim()).replace(/\d{1,3}$/,"0"); //network
between R1 and R2
var ip23 = (j.trim()).replace(/\d{1,3}$/,"0"); //network
between R2 and R3
var ip3client = (l.trim()).replace(/\d{1,3}$/,"0"); //network
ip between R3 and client
var ip3client2 = ip3client;
 document.getElementById("r11ip").innerHTML = a;
 document.getElementById("r11mask").innerHTML = b;
 document.getElementById("r12ip").innerHTML = c;
 document.getElementById("r12mask").innerHTML = d;
 document.getElementById("r21ip").innerHTML = e;
```

```
 document.getElementById("r21mask").innerHTML = f;
 document.getElementById("r22ip").innerHTML = g;
 document.getElementById("r22mask").innerHTML = h;
 document.getElementById("r31ip").innerHTML = j;
document.getElementById("r31mask").innerHTML = k;
 document.getElementById("r32ip").innerHTML = l;
 document.getElementById("r32mask").innerHTML = m;
 document.getElementById("r21ip2").innerHTML = ee;
 document.getElementById("r21ip3").innerHTML = eee;
 document.getElementById("r12ip2").innerHTML = aa;
  document.getElementById("r32mask2").innerHTML = mm;
  document.getElementById("r31ip2").innerHTML = jj;
  document.getElementById("r31mask2").innerHTML = kk;
  document.getElementById("r32mask3").innerHTML = mmm;
  document.getElementById("r11mask2").innerHTML = bb;
  document.getElementById("r22ip2").innerHTML = gg;
 document.getElementById("r22ip3").innerHTML = ggg;
 document.getElementById("r11mask3").innerHTML = bbb;
 document.getElementById("servr1net").innerHTML = ipserv1;
 document.getElementById("r1r2net").innerHTML = ip12;
 document.getElementById("r2r3net").innerHTML = ip23;
 document.getElementById("r3clientnet").innerHTML =
ip3client;
 document.getElementById("ip3clientnet2").innerHTML =
ip3client2;
 document.getElementById("servip").innerHTML = serv;
 document.getElementById("rev32mask").innerHTML =
RevertMask(m);
```

```
}
function RevertMask(mask) {
if (mask == "255.255.255.0")
{
    var str = mask.split('.');
    var reverse = "0.0.0."+str[0];
}
if (mask == "255.255.0.0")
{
    var str = mask.split('.');
    var reverse = "0.0."+str[1]+"."+str[0];
}
if (mask == "255.0.0.0")
{
    var str = mask.split('.');
var reverse = "0."+str[2]+ "." +str[1]+ "." +str[0]
}
return reverse;
}
function CopyToClipboard1(containerid) {
 if (document.selection) {
 var range = document.body.createTextRange();

range.moveToElementText(document.getElementById(containerid))
;
 range.select().createTextRange();
 document.execCommand("copy");
```

```
 } else if (window.getSelection) {

 var range = document.createRange();
 range.selectNode(document.getElementById(containerid));
 window.getSelection().addRange(range);
 document.execCommand("copy");
 alert("Copied, paste in router")
 }
}
</script>
   </html>
```